Your search matched **2** of **807900** documents.
Results are shown **15** to a page, sorted by **publication year** in **descending** order.
You may refine your search by editing the current search expression or entering a new one the text box.

Then click **Search Again**.

appliance <and> control* <and> gesture

Search Again

Results:
Journal or Magazine = **JNL** Conference = **CNF** Standard = **STD**

1 **Rock 'n' Scroll is here to stay [user interface]**
Bartlett, J.F.
IEEE Computer Graphics and Applications , Volume: 20 Issue: 3 , May-June 2000
Page(s): 40 -45

[Abstract] [PDF Full-Text (640 KB)] **JNL**

09

2 **System architecture and techniques for gesture recognition in unconstrained environments**
Kohler, M.R.J.
Virtual Systems and MultiMedia, 1997. VSMM '97. Proceedings., International Conference on , 1997
Page(s): 137 -146

[Abstract] [PDF Full-Text (820 KB)] **CNF**

5,906

System Architecture and Techniques for Gesture Recognition in Unconstraint Environments

M. R. J. Kohler
Informatik VII (Computer Graphics)
University of Dortmund
44221 Dortmund, Germany
e-mail: Markus.Kohler@cs.uni-dortmund.de
http://ls7-www.informatik.uni-dortmund.de/~kohler

Abstract

Controlling appliances in home environments by gestures is a step towards more intuitive and natural human computer interfaces. A brief overview on an existing vision based gesture recognition system and its architecture and details on ergonomic remote control of devices by gestures are clarified. The focus is on the motion detection, object normalization and identification, the modelling and the prediction of motion by the Kalman Filter. The initialization problem of the Kalman Filter of a vision based system for human motion tracking differs from initialization for physical systems, where manuals report measurement errors. A main interest was to develop the initialization and adequate Kalman model for human motion. Most aspects mentioned in this report were implemented in the ARGUS prototype.

1 Introduction

Human computer interfaces based on visual input (video) have found growing interest during the last few years. One reason may be the continuously falling expense of hardware for image grabbing and processing. Even colour image processing is now available and fast enough for pattern recognition.

There are several examples of gesture recognition systems and tracking systems (survey [20]). A motion recognition system was presented by Darrell and Pentland [6] using special hardware. "Digit Eyes" [21] models hand postures with 27 degrees of freedom and tracks the motion of the fingers. Ahmad [1] describes a real time tracking system where 19 degrees of freedom are available. Maggioni [17, 18] enumerates several useful applications of a vision based interface. A colour look-up table is determined considering both, the object's and background colours. It is used for segmentation. Bröckl-Fox [3] considers ergonomic and anatomic aspects concerning the gesture based interaction between user and computer. An ergonomically remarkable "Television control by hand gestures" war presented by Freeman [10]. The problem of perspective distortion also affecting ARGUS was already discussed in [19, 7]. All these systems implement different input techniques and several aspects in gesture recognition in constraint environments.

This report discusses the research and underlying problems and solutions of the project ARGUS [13]. The goal of the prototype system ARGUS is to apply visual input in home environment without the need of markers attached to the user's body. The user may communicate with the devices in his home environment in a natural way. There are no constraints on the position from where he communicates through gestures as long as he is within the *interaction area*[1]. ARGUS shows that gestural input can be realized on low cost personal computers. This makes ARGUS available for disabled (e. g. speech impaired) and elderly people. The remote control in this prototype is done completely by gesture input, and it is a challenge to evaluate the usefulness of gestural input. In the near future, ARGUS will be combined with speech input because the combination of speech and visual input is the most natural way of communication: "Open that window" when pointing towards a window. Neither remote control by gesture recognition itself nor by mere speech recognition will work. Speech recognition fails in noisy environments (if the TV or audio devices are switched on) and gesture recognition can not be used in dark environments to switch on the light, for ex-

[1] Assuming that a finger appearing with two pixel thickness is good enough to be recognized continuously and a real finger has a thickness of 18 mm, an interaction area of 3.45 m diameter is derived with a maximum x resolution of the camera's colour channels U/V of 384 pixel. This is based on the assumption, that 5 fingers with 4 spaces between must be recognized.

0-8186-8150-0/97 $10.00 © 1997 IEEE

Figure 1: The above gesture sequence is called *pointer click*. The very left hand posture shows the *pointer gesture*.

ample. Visual and speech input will complete each other. The output of ARGUS is speech.

The following Section 2 describes the assembly, functionality and architecture of the ARGUS system. Section 3 explains details about motion detection, tracking and object identification and Section 4 outlines the results and future investigation.

2 Assembly and Functionality of ARGUS in an Home Environment

This Section shows briefly the overall assembly, functionality and system architecture of ARGUS. In the calibrated ARGUS system (see [13] for calibration) two cameras observing the user are installed close to the wall above the devices (Fig. 3). The user points towards a device with straddled thumb and forefinger. The system reports the aimed device by speech feedback. The selection is done by bowing the thumb (called *pointer-click*, Fig. 1). Further control is according to Table 1, where one gesture is used for each line. Every gesture is mapped to several similar tasks from different devices[2], which reduces the number of gestures and makes the dialogue more intuitive. Additional functionality is achieved by gesture sequences. Devices like a VCR, TV, CD player, lamps or coffee maker etc. (Fig. 3) may be connected to the computer either through a home bus system (separate cables, power supplies, radio waves — bidirectional), or they may be controlled with infrared signals (unidirectional) by the computer [13].

The ARGUS system architecture consists of the four recognition systems (Fig. 2), *Motion Recognition, Colour Recognition, Gesture Recognition and Sequence/Trajectory Recognition*, and the *Dialogue Control System*. For each *object* (hands or head) detected by the subsystem *Object Recognition and Tracking* (Section 3), one instance of the modules *Segmentation, ZYKLOP* and *Motion Interpreter* is generated.

The subsystem *Segmentation* separates the foreground from the background for each *object* separately in an almost natural environment based on a V channel histogram [13, 2, 14]. In a future step the prom-

[2]The user will communicate with one device at a time, only.

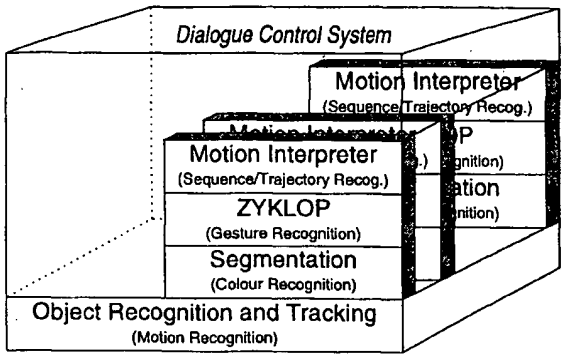

Figure 2: System architecture of the Integrated Gesture Recognition System

ising colour recognition from Schröter[3] [23] and further edge and motion based methods will be included, as the independence of variation in colour and brightness still needs to be improved in ARGUS.

A further segmentation step, called *automatic cutting*, separates or normalizes the hand objects. If the forearm is partly covered by clothes, it generally confuses gesture recognition. The resulting boundary adulteration is similar to the problem of recognizing occluded or overlapping objects. Hence, the referenced solutions proposed in literature [25, 11, 12, 16] will be further examined. Another method is to use a code sequence recognition like that implemented in ZYKLOP [24]. It may be applied to Chain Codes [5, 9] to recognize the relevant part of the contour (invariant to rotation, scaling and to a high degree to perspective distortion). Another promising method is to fit a growing circle (tracked by Kalman Filter) — called bubble —, that completely lies within the hand, (Fig. 5), which is similar to search for crossings in the skeleton. The search direction for this bubble may be forced by a potential function.

As a hand mainly "rotates" around the elbow the centrifugal force is an appropriate potential function to shift the bubble towards the end of the arm. The potential function and bubble shape depend on the object that needs to be separated. If a head needs to be separated from the body, a head detecting bubble should be driven by impetus forces and should be an ellipse, as a head is generally upright. Bubbles with search-object specific attributes may be used to separate body parts. If one of the above automatic cutting

[3]Schröter includes a survey of the available contributions concerning colour and brightness constancy.

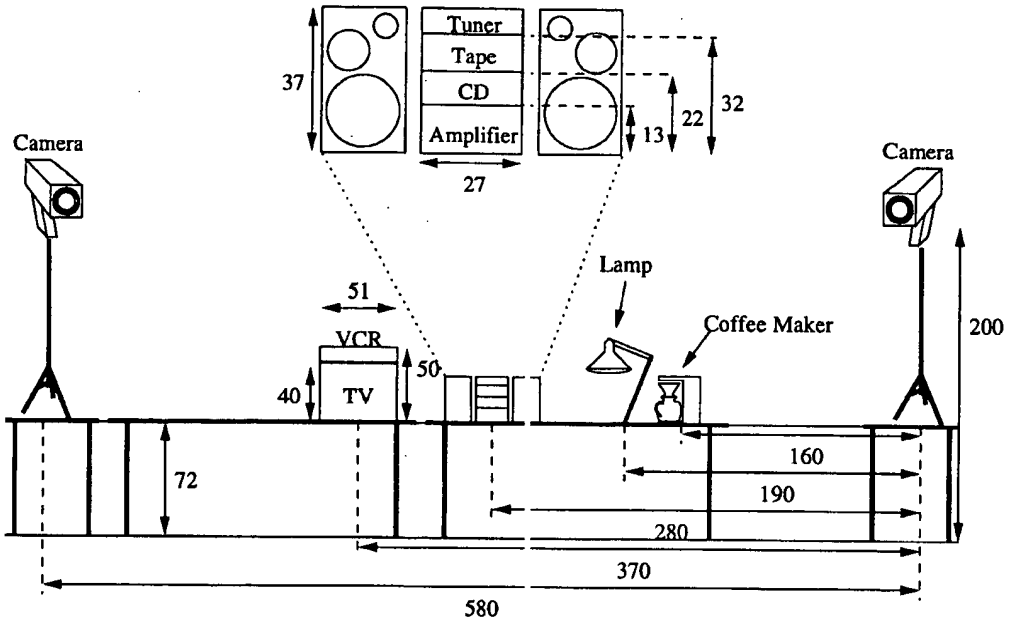

Figure 3: Test environment: Vision based remote control in intelligent home environments: (The measurement is in *cm*).

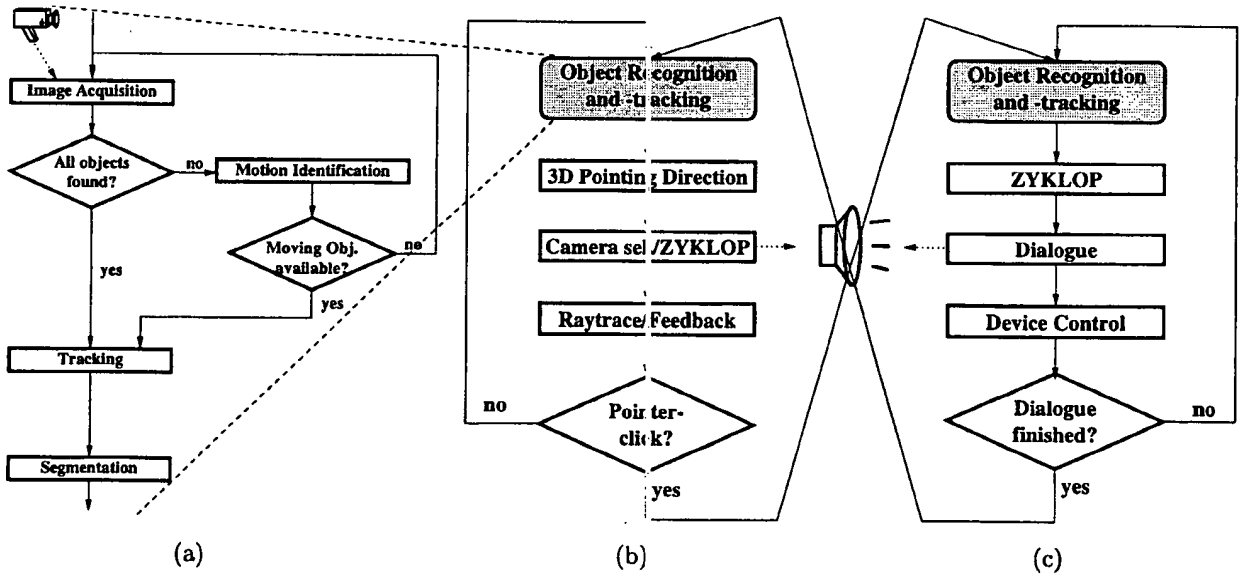

Figure 4: Control flow of the *Object Recognition and Tracking* (a), device selection by pointing including feedback *(Determine Direction and Device)* (b) and dialogue with one device *(Dialogue, Device Control)* (c).

Function	amplifier	compact disk player	tuner	compact cassette system	video cassette recorder	television	gesture
Volume +	⋆					⋆	
Volume −	⋆					⋆	
Play		⋆		⋆	⋆		
Stop		⋆		⋆	⋆		
Source	⋆						
Power off	⋆				⋆	⋆	
		title	program	wind	wind	program	
+ function		⋆	⋆	⋆	⋆	⋆	
− function		⋆	⋆	⋆	⋆	⋆	

Table 1: Example of common functions of devices: Same functions are mapped to the same gesture; similar functions may be mapped to the same gesture if this is intuitive and no other function is overloaded.

Figure 5: Automatic cutting: Separating the hand from the forearm by potential driven search bubbles.

methods is used, the "cut" gestures must be taught to the gesture recognition system.

Next, the pure gesture recognition [24] evaluates only the hand's bounding box of that camera, whose cosine of the angle between the optical axis and the pointing direction is minimal. This saves computing time. A meaningful gesture is further processed by the *Motion Interpreter* together with position parameters delivering the final gesture number. The subsystem Motion Interpreter includes the *Sequence and Trajectory Recognition* for each object. This was already included in ZYKLOP [24]. The sequence recognition may further be improved by using HMMs [26].

Finally, the Dialogue Control System combines a final gesture of the multiple output of the recognition process instances and decides, which task to complete and which device to control. The stereo view is used to estimate 3D pointing directions [13] and determining the selected device by a ray tracing step. Speech feedback compensates the inaccurate human pointing. Exchanging the Dialogue Control System by a simple interpreter table, that serves to translate gesture codes to commands, enables to use ARGUS as a single camera gesture recognition system for controlling desktop applications through ASCII commands (see project ZYKLOP [24]) or X events.

Fig. 4 shows the detailed, self explaining control flows of ARGUS for the main system states: (a) object recognition and tracking (continues during both processes (b) and (c)), (b) device selection by pointing and (c) dialogue with one device — either (b) or (c) is active.

3 Object Recognition and Tracking

The subsystem *Object Recognition and Tracking* is able to handle up to three cameras with arbitrary moving objects and includes the basic modules *Image Acquisition, Image Management* and *Bounding Box Management* [13]. In the following subsections we discuss the special topics: an improved *Motion Detection* compared to [22], *Tracking* including the initialization and the *object identification*.

3.1 Motion Detection

The module *Motion Detection* uses the difference images for each camera and each channel. Due to noise of the camera, only differences exceeding a certain threshold are considered. A fast filter algorithm detects the relevant motion according to the ideas of the *Motion to Shape Sensor* [22]. It first operates on lines of the difference image. Each finger must appear at least as one pixel to be available for hand gesture recognition. Further, gesture recognition requires that spaces between straddled fingers are visible. Hence, hand objects must result in run lengths with at least five pixels, where longer run lengths may contain up to approximately 45 % non-set pixels. All the run lengths of that format are accepted and otherwise rejected. This rule already eliminates most of the noisy motion. Next coherence is examined by a scan line algorithm. Bounding boxes are generated for the detected domains. A sequence of the following heuristic filters may be configured in ARGUS and the order is essential to the result. Filters may be called several times. Some filters are (see also [13]): If bounding boxes are not tall enough (at least five pixel), they are removed. Boxes with less distance, than a certain percentage of the average distance between all the boxes, are melted together. Too large, too small and very eccentric boxes are eliminated. Experiments showed that these filters mostly eliminate all the noise but no relevant objects. The resulting sensor is very sensitive to motion. An advantage is the speed up, because subsampling of the image can easily be applied. All the detected moving objects (resp. labeled bounding boxes) are tracked according to the next section.

3.2 Tracking

The control flow of ARGUS object recognition and tracking is shown in the self explaining Fig. 4 (a). Motion Detection is called interchangeably with Tracking. Hence, once a visible object is detected it will not get lost even when it stops moving. The *Tracking Module* uses the *Discrete Kalman Filter* [4]. The filter predicts the position of each object's bounding box in the two dimensional image.

The physical law of motion $\mathbf{s}(t) = \mathbf{s}(0) + \mathbf{v}(0) \cdot t + 1/2 \cdot \mathbf{a}(0) \cdot t^2$ results from truncating the Taylor Series of a time variant motion. If acceleration is constant, it exactly models the motion. If velocity is constant, then $\ddot{\mathbf{s}}(t) = \dot{\mathbf{v}}(t) = \mathbf{a}(t)$ is zero and the constant velocity model follows $\mathbf{s}(t) = \mathbf{s}(0) + \mathbf{v}(0) \cdot t$. However, if neither $\mathbf{v}(t)$ nor $\mathbf{a}(t)$ is constant, both laws only hold for very small t, due to generally bad convergence of

Taylor Series. What does it mean, that the acceleration is constant? It means that the force F driving the object of mass m is constant, because $F = m \cdot a$, with acceleration a. This may be true for vehicles and missiles after some startup time. But in human motion forces keep changing continuously.

Thus, the motion is considered to be the superposition of an ideal basic motion of constant velocity, for example, and the acceleration as white noise. For further information about "white noise" signals we refer to [15].

3.2.1 Discrete Time Model of the Kalman Filter

We start to model a motion of a single point with the Kalman Filter (e. g. the lower left, upper right corner of a bounding box or the center of gravity)[4].

Experiments with ARGUS showed that modelling human motion with constant velocity and considering acceleration as white noise is sufficient. If we idealize the human motion to the constant velocity model, i. e. we assume the velocity is constant in the time interval $[t_k, t_{k+1}]$, we can transform it to a description with linear equations. More generally, let $\mathbf{s}_j := \mathbf{s}(t_j)$ be a position of an object at time t_j and $\mathbf{v}_j := \mathbf{v}(t_j)$ its velocity. Then it is $\mathbf{s}_{k+1} = \mathbf{s}_k + \mathbf{v}_k \cdot (t_{k+1} - t_k)$ and $\mathbf{v}_{k+1} = \mathbf{v}_k$, which is equivalent to

$$\begin{pmatrix} \mathbf{s}_{k+1} \\ \mathbf{v}_{k+1} \end{pmatrix} = \Phi_\mathbf{k} \cdot \begin{pmatrix} \mathbf{s}_k \\ \mathbf{v}_k \end{pmatrix}, \quad \Phi_\mathbf{k} := \begin{pmatrix} 1 & \Delta t \\ 0 & 1 \end{pmatrix} \quad (1)$$

and $\Delta t := t_{k+1} - t_k$. The vector $\mathbf{x}_k := (\mathbf{s}_k, \mathbf{v}_k)^T$ is called the *process state vector* at time t_k. The matrix $\Phi_\mathbf{k}$ is the *transition matrix* that relates $\mathbf{x}_k$ to $\mathbf{x}_{k+1}$ in absence of a forcing function. However, human motion is not constant at all. It is driven by arbitrary forces and the acceleration seems to be random. Therefore we model this aspect by a random noise $\mathbf{w}_k$, which is added to the left side of Equation (1) yielding

$$\mathbf{x}_{k+1} = \Phi_\mathbf{k} \cdot \mathbf{x}_\mathbf{k} + \mathbf{w}_\mathbf{k} \,. \quad (2)$$

For detailed explanation see [15]. The time between two images is assumed to be constantly Δt.

During the recursion of the Kalman Filter each prediction will be followed by a measurement. Putting this measurement into the recursion will influence the next prediction. This is best described in [8, 15]. So the Kalman Filter "learns" the data included by the measurement. The measurement of the random process occurs in time at discrete points in accordance with the linear relationship

$$\mathbf{z}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{e}_k \quad (3)$$

where $\mathbf{z}_k$ is the *vector measurement* and $\mathbf{H}_k$ gives the ideal (noiseless) connection between the measurement and the state vector at time t_k. Hence, the Kalman Recursion considers that the internal state vector of the model may not directly be measured.

In our case, we can only measure positions on the screen in two dimensions. We cannot measure velocity but only calculate it roughly from several positions. But therefore we use the Kalman Filter. The state vector is modelled to have a position and a velocity. The matrix $\mathbf{H}_k$ shall only extract the position from the state vector which is done by $\mathbf{H}_k = (1, 0)$. Further, $\mathbf{e}_k$ is the *measurement error* and also should have white characteristic. Let $\mathbf{z}_k$ be the measurement, then $\mathbf{z}_k = \mathbf{s}_k + \mathbf{e}_k$, i. e. the measurement $\mathbf{z}_k$ is identical to the position $\mathbf{s}_k$ of the process state vector plus a measurement error $\mathbf{e}_k$.

Physically, forces that induce acceleration and velocities do not influence each other, if they are perpendicular (superposition of orthogonal forces). Any translational force, acceleration and velocity in 2D or 3D space is a superposition of two resp. three orthogonal forces, accelerations or velocities. Hence, the Kalman Model (2) and the measurement relation (3) may be written for each dimension separately [15]. Thus, for two dimensional prediction as in the ARGUS case two separate Kalman recursions are evaluated. If several objects in the plane are tracked, it is highly recommended to split up the system for reasons of time efficiency of matrix inversion.

3.2.2 Initialization of the Filter Covariance Matrices

The Kalman Recursion requires some covariance matrices, that must be determined and initialized: the covariance matrix of the measurement error $\mathbf{R}_k$, the covariance matrix of the white acceleration $\mathbf{Q}_k$ and the covariance matrix of the estimation error $\mathbf{P}_k$. Further, the a priori estimate $\hat{\mathbf{x}}_0^-$ must be set to an initial value. This initialization is discussed in detail in [15]. Here the results are summarized. Because gesture recognition is driven near the lower resolution boundary to achieve a bigger interaction area, it is a prerequisite,

[4] As motion prediction is applied to images, the positions, velocities etc. will have two dimensions. But let us abstract from our special situation. The values $\mathbf{s}(t), \mathbf{v}(t), \mathbf{a}(t)$ might be a vector or a scalar. The fact, that $\mathbf{s}(t)$ might be a vector, is marked by bold letters. All matrices and vectors are also marked bold and matrices are, in contrary to vectors, in capitals. All equations should be regarded carefully, as they might imply tensor products.

that the measurement error should be below one pixel. Further, the x and y measurement error are independent. According to [15], this leads to the initialization of the *measurement error covariance matrix*

$$\mathbf{R}_k = \frac{2}{3} \cdot \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \forall k \,,$$

Note, that this matrix is related on the unit "pixel", because measurement is done in pixels. With $\Delta t = t_{k+1} - t_k$ as previously defined, [15] inspires to initialize the covariance matrix of the white acceleration to

$$\mathbf{Q}_k = \frac{a^2 \Delta t}{6} \begin{pmatrix} 2\mathbf{I}\Delta^2 t & 3\mathbf{I}\Delta t \\ 3\mathbf{I}\Delta t & 6\mathbf{I} \end{pmatrix} s$$

while a is the spectral amplitude of the white noise and $\mathbf{I}$ is the 2×2 identity (s is the unit seconds). As the acceleration is identified as white noise, a is the amplitude of the acceleration frequencies. According to experiments we estimate the spectral amplitude to a value smaller than $a \approx 1225 \frac{pixel}{s^2}$ for fast hand movements (the assumption was, that the motion mainly takes place in the maximum interaction area of 3,45 m), which is the same as $a = 49$ pixel/frame^2 at 5 frames per second, $a = 12$ pixel/frame^2 at 10 frames per second or $a = 5$pixel/frame^2 at 15 frames per second.

At this point we assume that we have an initial estimate of the process at some point in time t_k, and that this estimate is based on all of our knowledge about the process prior to t_k. This prior (or *a priori*) estimate will be denoted as $\hat{\mathbf{x}}_k^-$ where the "hat" denotes estimate, and the "super minus" is a reminder that this is our best estimate prior to assimilating the measurement at t_k. The estimation error is then $\mathbf{x}_k - \hat{\mathbf{x}}_k^-$ and the associated error covariance matrix $\mathbf{P}_k^- = E[(\mathbf{x}_k - \hat{\mathbf{x}}_k^-)(\mathbf{x}_k - \hat{\mathbf{x}}_k^-)^T]$. According to [15] we take the covariance matrix of the estimation error as

$$\mathbf{P}_0^- = \frac{1}{16} \cdot \begin{pmatrix} s^2 & 0 \\ 0 & v^2 \end{pmatrix} .$$

The report [15] suggests to take the distance $s = 25$ pixel at 5 frames per second, $s = 6$ pixel at 10 frames per second and $s = 3$ pixel at 15 frames per second and the maximal velocity $v = 49$ pixel/frame at 5 frames per second, $v = 12$ pixel/frame at 10 frames per second or $v = 5$ pixelframe at 15 frames per second. Again this suggestion was made under the assumption that the action mostly takes place such that the hands appear near the resolution boundary. The initial value of the process state vector is set to $\hat{\mathbf{x}}_0^- = (\mathbf{s}_0^-, 0)^T$, where $\mathbf{s}_0^-$ is the center of gravity of the (two!) domains of an object detected by the motion detection.

3.2.3 The Kalman Recursion

The matrices $\mathbf{P}_0^-, \mathbf{R}_0$ and $\mathbf{Q}_0$ must be initialized previously as well as the $\hat{\mathbf{x}}_0^-$. It is important to find good values for $\mathbf{R}_0$ and $\mathbf{Q}_0$. If the values $\hat{\mathbf{x}}_0^-$ or $\mathbf{P}_0^-$ are poor they will mostly converge towards the expected values. Next the *Kalman Recursion* is evaluated (see also [4, pages 195–200] and [15])

1. Compute gain (blending factor)
   $\mathbf{K}_k = \mathbf{P}_k^- \mathbf{H}_k^T (\mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T + \mathbf{R}_k)^{-1}$
2. Update estimate $\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + \mathbf{K}_k(\mathbf{z}_k - \mathbf{H}_k \cdot \hat{\mathbf{x}}_k^-)$
3. Update *error covariance matrix*
   $\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)\mathbf{P}_k^-$
4. Project ahead $\hat{\mathbf{x}}_{k+1}^- = \mathbf{\Phi}_k \hat{\mathbf{x}}_k$
5. Update *error covariance matrix*
   $\mathbf{P}_{k+1}^- = \mathbf{\Phi}_k \mathbf{P}_k \mathbf{\Phi}_k^T + \mathbf{Q}_k$.

The blending factor and the update estimate is discussed in [15]. For each relevant bounding box a new position is estimated with the Kalman Filter. The main advantage of Kalman Filtering is that tracking speeds up the image processing and improves the independence of varying illumination. Further advantages are reported in [13].

To make sure that a tracked, fast moving object (like the hand) completely lies within the predicted bounding box of the Kalman Filter, this box is increased by a certain factor. Further, if two objects partly overlap, the bounding boxes may coincide after segmentation, as the objects appear as a single object. In this case if two objects are detected within one box that emerged from previously two boxes, the estimated box must be split to two boxes surrounding the two detected objects.

3.3 Object Identification

Another part of the Object Recognition and Tracking subsystem is the *Object Identification*. It is important to gradually identify the objects. If an object (motion) is detected, it will be tracked immediately and it is first identified to a certain confidence by its absolute and relative position to the other objects, by its size, eccentricity, colour and further heuristics. For further stabilization or correction of the identification, the process state vector of the tracking process is used, which includes motion parameters as the position and velocity and may be extended by the acceleration and further derivatives. Various features are extracted from the process state vector. These features are compared to previously learned ordinary human motion parameters of hands and heads. As

head movements are generally of slower translational acceleration compared to hand movement, it is possible to identify objects by motion parameters.

As the reliability of the identification increases with the amount of time an object is tracked, the confidence attribute of the object's identifier must be increased. The object must be continuously compared against other objects in all camera views: firstly, to skip it as soon as possible, if it is identical to another tracked object in the same camera view[5], and secondly, to recognize equal objects in different camera views and give them the same identifier. Thus even the left and the right hand may be distinguished as they will have similar motion characteristic in both projections. In case of uncertainty the heuristics (like relative/absolute position, size etc.) should further be considered.

4 Results and Future Improvements

ARGUS showed that it is possible to use gestures for remote controlling devices in a home environment. The functionality was evaluated in a test environment illustrated in Fig. 3 and Fig. 6. In the ARGUS project a PC Pentium 133 MHz with 16 MByte was used. The performance was ranging from 5 to 15 frames per second and camera [13]. It turned out that rates below 10 frames per second and camera appear to slow for a natural and comfortable dialogue. The angular resolution is below one degree.

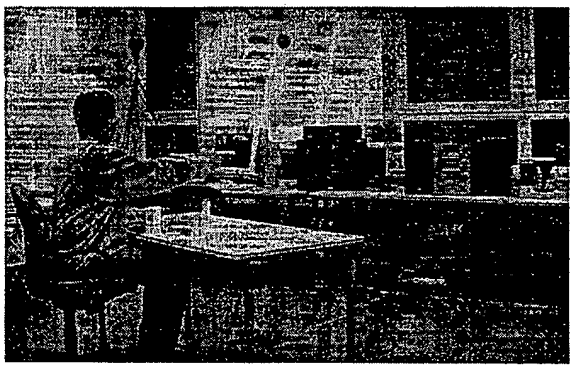

Figure 6: ARGUS labs: Remote controlling audio and video devices by gestures.

Further improvements will be the inclusion of colour recognition according [23] to be more independent from illumination, the use of rotatable cameras to increase the interaction area and the forearm segmentation. ARGUS must further be improved with respect to ergonomic aspects. The gesture dialogue must be improved by selecting easily performable gestures, by adapting timeouts and latency timers to the user needs and by an overall intelligent speech and gesture dialogue interface.

[5]This might occur, because Tracking and Motion Detection are called interchangeably. So Motion Detection will also find the tracked objects.

Acknowledgments

I appreciate the great work that was done by the graduate students M. Breig, D. Dechow, M. Fellenberg, R. Jaeschke, M. Jedamzik, T. Kuleßa, D. Pukropski, D. Rief, D. Rosemann, A. Thümmler, J. Albrecht, D. Aust, L. Cleff, B. Deimel, C. Esken, C. Kirstein, M. Kukuk, S. Mork, S. Schewe, S. Schröter, G. Veltmann, C. Wenger and my colleagues M. Stark and M. Wittner. Thanks to Prof. Dr. H. Müller who supported both projects.

References

[1] Subutai Ahmad. A usable real-time 3D hand tracker. In *Proceedings 28th Asilomar Conference on Signals, Systems and Computers*, pages 1257–1261. IEEE Computer Society Press, 1995.

[2] Projektgruppe 277: ARGUS. Ein ergonomisches Dialogsystem zur Steuerung von technischen Systemen in Wohnbereichen mittels Gestenerkennung — Abschlußbericht. Technical report, Informatik VII, University of Dortmund, 1997.

[3] Ulrich Bröckl-Fox. *Untersuchung neuer, gestenbasierter Methoden für die 3D-Interaktion.* PhD thesis, University of Karlsruhe, February 1995. Shaker-Verlag, ISBN 3-8265-0620-0.

[4] Robert G. Brown. *Introduction to random signal analysis and Kalman filtering.* Wiley, New York [u.a.], 1983.

[5] Mo Dai, Pierre Baylou, and Mohamed Najim. An efficient algorithm for computation of shape moments from run-length codes or chain codes. *Pattern Recognition*, 25(10):1119 – 1128, February 1992.

[6] T.J. Darrell and A.P. Pentland. Space-time gestures. In *Proc. Conf. on Computer Vision and Pattern Recognition*, pages 335–340, New York, NY, June 15-17 1993.

[7] A. Drees, F. Kummert, E. Littmann, S. Posch, H. Ritter, and G. Sagerer. A hybrid system to detect hand orientation in stereo images. In E.S.

Gelsema and L.N. Kanal, editors, *Pattern Recognition in Practice IV: Multiple Paradigms, Comparative Studies and Hybrid Systems*, pages 551–562. Elsevier, Amsterdam, 1994.

[8] Roger M. du Plessis. Poor Man's Explanation of Kalman Filtering or how I stopped worrying and learned to love Matrix Inversion. Technical report, Autonetics Division, Rockwell International, 3370 Miraloma Avenue, Anaheim, California 92803, June 1967.

[9] H. Freeman. Computer processing of line drawing images. *Computing Surveys*, 6:57–98, 1974.

[10] W. T. Freeman and C. Weissman. Television control by hand gestures. In M. Bichsel, editor, *Proc. Intl. Workshop on Automatic Face- and Gesture-Recognition*, IEEE Computer Society, pages 179–183, Zürich, Switzerland, June 1995. http://www.merl.com/TR/TR94-24.

[11] W. Haettich. Recognition of overlapping workpieces by model directed construction of object contours. *DSIA*, 1(2–3), 1982.

[12] M. H. Han and D. Jang. The use of maximum curvature points for the recognition of partially occluded objects. *Pattern Recognition*, 23:21–33, 1990.

[13] Markus Kohler. Vision Based Remote Control in Intelligent Home Environments. In B. Girod, H. Niemann, and H.-P. Seidel, editors, *3D Image Analysis and Synthesis'96*, pages 147–154, University of Erlangen-Nuremberg/Germany, November 18–19 1996. infix Verlag. ISBN 3-89601-000-X; see also *Ein ergonomisches Dialogsystem zur Steuerung von technischen Systemen in Wohnbereichen mittels Gestenerkennung — Abschlußbericht* at Dep. f. CG/University of Dortmund.

[14] Markus Kohler. Technical details and ergonomical aspects of gesture recognition applied in intelligent home environments. Technical Report 638, Informatik VII, University of Dortmund, January 1997.

[15] Markus Kohler. Using the Kalman Filter to track human interactive motion — Modelling and initialization of the Kalman Filter for translational motion —. Technical Report 629, Informatik VII, University of Dortmund, January 1997.

[16] H. C. Liu and M. D. Srinath. Partial shape classification using contour matching in distance transformation. *IEEE Trans. Pattern Analysis Mach. Intell. PAMI-12*, 11:1072–1079, 1990.

[17] Ch. Maggioni. GestureComputer: New ways of operating a computer. In M. Bichsel, editor, *Proc. Intl. Workshop on Automatic Face- and Gesture-Recognition*, IEEE Computer Society, pages 166–171, Zürich, Switzerland, June 1995.

[18] Ch. Maggioni and B. Kämmerer. GestureComputer – history, design and applications. In Ed. R. Cippola and A. Pentland, editors, *ECCV '96 Workshop on Computer Vision in Man-Machine Interfaces*. Cambridge University Press, to appear in June 1996.

[19] Andrea Meyering and Helge Ritter. Learning to recognize 3d-hand postures from perspective pixel images. *Artificial Neural networks*, pages 821–824, 1992. Elsevier Science Publishers B.V.

[20] Thomas S. Huang Vladimir I. Pavlovic. Hand gesture modeling, analysis and synthesis. In M. Bichsel, editor, *Proc. Intl. Workshop on Automatic Face- and Gesture-Recognition*, IEEE Computer Society, pages 73–79, Zürich, Switzerland, June 1995.

[21] James M. Rehg and Takeo Kanade. DigitEyes: Vision-based human hand tracking. Technical report, School of Computer Science, Carnegie Mellon University, December 1993.

[22] Hans Röttger. Echtzeit-Interaktion auf Basis visueller Bewegungserfassung. Master's thesis, University of Dortmund, July 1995.

[23] Sven Schröter. Entwicklung von Verfahren zur automatischen Kalibration eines farbbasierten Objekterkennungssystems. Master's thesis, University of Dortmund, August 1996.

[24] Michael Stark and Markus Kohler. Videobasierte Mensch-Maschine-Interaktion. *Informationstechnik und Technische Informatik*, 38(3):15–20, June 1996. More Details in english *Video Based Gesture Recognition for Human Computer Interaction*, TR 593 and *Projektgruppe 247: ZYKLOP — Visueller Mensch-Rechner-Dialog — Abschlußbericht (1995)* at Dep. f. CG/University of Dortmund.

[25] Peter W.M. Tsang, P.C. Yuen, and F.K. Lam. Recognition of occluded objects. *Pattern Recognition*, 25(10):1107–1117, 1992.

[26] Junji Yamato, Jun Ohya, and Kenichiro Ishii. Recognizing human action in time-sequential images using hidden markov model. *IEEE*, pages 379–385, 1992. ISBN or ISSN 0-8186-2855-3/92.